IN THE CLAIMS:

1. (currently amended) A system including a multi-tier application architecture having a middletier, said system comprising:

a graphical user interface; and

a framework to mediate between an application within a front-end tier and the middletier, wherein the framework is configured to:

allow the middletier to execute an object fetched by the application from a cache;

when the execution of the object fails, repeatedly refresh the object within a limited number of retries;

when the object refresh succeeds, return the object to the cache and again allow the middletier to execute the object; [[and]]

when the object refresh does not succeed within the limited number of retries, quit the application in a fail-safe way; and

output an operational state of the framework using the graphical user interface.

2. (currently amended) The system according to claim 1, wherein the framework is configured to allow a user to specify the limited number of retries via the graphical user interface.

3. (currently amended) The system according to claim 2, wherein the framework is configured to allow the user to specify a time interval between the retries via the graphical user interface.

4. (currently amended) The system according to claim 1, wherein the framework operations are visible to a user via the graphical user interface.

5. (previously presented) The system according to claim 1, further including a watchdog configured to resume normal operations when the middletier crashes.

6. (previously presented) The system according to claim 5, wherein the watchdog is configured to recover the middletier based on a result of periodical polling.

7. (previously presented) The system according to claim 5, wherein the watchdog is configured to recover the middletier based on notification from the framework.

8. (previously presented) The system according to claim 1, wherein the framework comprises a logic controller, a detector, a refresher, and a quitter.

9. (currently amended) A method of executing an application, said method comprising:

transmitting an object used by the application within a first tier to a second tier;

executing a logic program at the second tier, wherein the logic program corresponds to the transmitted object;

detecting an execution status of the logic program at the first tier, said detecting comprising:

detecting when the execution of the logic program fails such that the object becomes stale;

repeatedly refreshing the object within a limited number of retries; [[and]]

if said refreshing succeeds, then returning the object to the first tier and transmitting a second object to the second tier from the first tier; and

if said refreshing does not succeed within the limited number of retries, then quitting the application in a fail-safe way.

10. (currently amended) [[A]] The method in accordance with Claim 9 wherein transmitting an object used by the application further comprises transmitting the object from a cache within the first tier to the second tier.

11. (currently amended) [[A]] The method in accordance with Claim 10 wherein transmitting an object from a cache further comprises transmitting the object from the cache through a framework within the first tier to the second tier.

12. (currently amended) [[A]] The method in accordance with Claim 9 wherein detecting an execution status of the logic program at the first tier further comprises detecting an execution status of the logic program at a framework within the first tier.

13. (currently amended) [[A]] The method in accordance with Claim 9 further comprising, when the second tier crashes, resuming normal operation using a watchdog.

14. (currently amended) [[A]] The method in accordance with Claim 13 wherein resuming normal operation further comprises resuming normal operation based on periodical polling of the second tier.

15. (currently amended) [[A]] The method in accordance with Claim 13 wherein resuming normal operation further comprises recovering the second tier based on notification from a framework within the first tier.

16. -- 20. (cancelled)